1D-mosaics grouping using lattice vector quantization for a video browsing application

Vincent Ricordel, Florent Autrusseau, William Dupuy, Dominique Barba

1D-MOSAICS GROUPING USING LATTICE VECTOR QUANTIZATION FOR A VIDEO BROWSING APPLICATION

Vincent Ricordel, Florent Autrusseau, William Dupuy and Dominique Barba

Image and Video Communication Team, IRCCyN UMR CNRS 6597
Ecole Polytechnique de l'Université de Nantes
La Chantrerie, BP 50609, 44306 Nantes Cedex 3
{vincent.ricordel, florent.autrusseau, william.dupuy, dominique.barba}@polytech.univ-nantes.fr

ABSTRACT

1D-mosaics have been introduced as a tool for structuring and navigation in video content. These objects can be considered as the spatio-temporal signatures of the video shots. Our work aims at grouping automatically the video shots into scenes using these signatures. The original method is based on the tree-structured lattice vector quantization of the 1D-mosaics. Because of the hierarchical structure of the code-books, they can be compared progressively, and lattice use is time efficient. Indexing retrieval results are given for two video sequences, and different mosaics are successively compared to each other in order to assess the presented scheme's effectiveness.

1. CONTEXT OF THE WORK

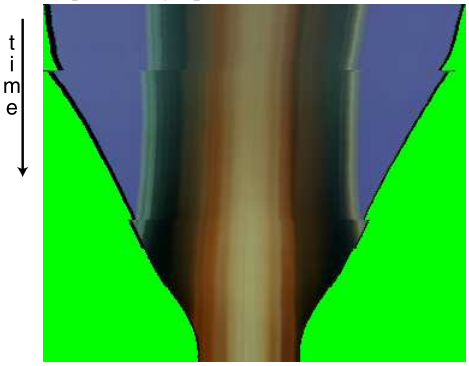
The work takes place in the context of tools design for structuring and browsing through digital video documents.
The storage and the search of documents in multimedia information systems assume that structuring and indexing of videos have been done before. MPEG7 [1] requires a structural decomposition of video documents into segments wich can be represented as semantic entities: objects, shots, scenes. Indexing should be done to characterize their content, movement, color, texture. The standard defines the video segment descriptors but not the way of performing the structural description. Moreover the future success of multimedia services of home intelligent devices depends on a neat navigation system. A clever structuring of video is essential to a good navigation.
Research in the field of video document structuring assumes a hierarchical decomposition of documents which shows such structural units as "scene", "shot", "object" or "event". We propose in this paper a method to structure video documents by grouping shots into scenes for browsing, from a navigation interface, through the video. Scenes are defined here as grouping of shots which share common color set. Spatio-temporal maps (or mosaics) have been introduced [2]. These static images give a global view of a video scene inside one shot, they are especially attractive in case of panoramic camera motions. We focus on 1-D mosaics which integrate spatio-temporal images and can play the role of spatio-temporal color signatures of video shots. Exactly we use them for clustering video segments into higher order entities such as video scenes.

2. 1-D MOSAIC

An image can be represented by a set of "1-D projections" using a Radon-like projective transform called Mojette projection [3] and integrating image signal along direction defined by a given angle. For each direction of projection, an 1-D mosaic can be built, interested readers may refer to [4] and [5] for more details on the 1D-mosaics creation. It corresponds to a "spatio-temporal" 1-D signal. The latter is obtained by motion-based compensation of projections of all the frames of the video sequence, into the coordinate system associated with the projection of a specific reference frame. Hence in a video sequence, motion has to be estimated first for mosaic construction.
Affine models of apparent motions in 2-D image plane have proved to be interesting in the characterization of typical forms of video shootings. We use a 3 parameters motion model with shift and zoom factor despite its incompleteness, as it allows for characterization of most frequent situations in video such as pan, traveling, "zoom in" and "zoom out". This motion model is specifically in the focus of our attention, as there exists a simple relationship between the 2-D motion model in image plane and 1-D motion model in the 1-D projective Radon transform domain we use for 1-D representation of video.
For a given shot, by using the motion parameters, we compensate each 1-D projection into the referential of the first 1-D projection (which corresponds to the first image of the shot). Note however that although this first image of the shot is taken as a reference, the whole frames in the considered

shot are used while computing the 1D-mosaic. We show in figure 1(a) some frames of a shot taken from the video sequence "Tympanon" (SFRS, "Service du Film de Recherche Scientifique"). In figure 1 (b) each line of the picture represents a motion-based compensated 1-D projection into the referential of the top 1-D projection. Then to get the final 1-D mosaic, each column of this picture is reduced using a median filter. Figure 1(c) shows the corresponding 1-D mosaic. We can see that the 1-D mosaic is like a color signature of the shot. Thus, from these color signatures, our goal becomes to group shots into scenes.

(a) Some frames of a shot (zoom in) of the sequence "Tympanon" (SFRS)



(b) Motion-based compensated 1D projections

(c) The corresponding 1-D mosaic

Fig. 1. Example of a 1D-mosaic construction.

3. TREE-STRUCTURED LATTICE VECTOR QUANTIZATION

3.1. Vector Quantization

VQ (from now VQ means vector quantizer as well as vector quantization) has been investigated extensively [6]. Let $x(n)$: $\mathbf{x} = (x(1), \ldots, x(k))$ be a k-component source vector. A VQ of dimension k and size L is defined as a function that maps a vector $\mathbf{x}$ into one of L reproduction vectors $\mathbf{y}_1, \ldots, \mathbf{y}_L$ belonging to $\mathbb{R}^k$. The set of reproduction vectors or codewords, is the code-book. This chart implicitly determines a partition of $\mathbb{R}^k$ in L non-overlapping Voronoï regions C_i typically defined by the equation (where $L_p(\mathbf{x}) = \sum_{i=1}^{k} |x_i|^p$ is the norm):

$$C_i = \left\{ \begin{array}{l} \mathbf{x} \in \mathbb{R}^k; \mathcal{Q}(\mathbf{x}) = \mathbf{y_i}, \\ \text{if } L_2(\mathbf{x} - \mathbf{y}_i) \leq L_2(\mathbf{x} - \mathbf{y}_j), \forall j \neq i \end{array} \right\}$$

Thus a VQ combines both functions: an encoder that, from the input vector $\mathbf{x}$, generates the index i specified by $\mathcal{Q}(\mathbf{x})$; a decoder that, from this index and by listing the code-book, generates the corresponding reproduction vector $\mathbf{y}_i$. In a transmission or storage context, a binary word or index is assigned to each codeword $\mathbf{y}_i$, and the goal consists in, for a given rate (resp. distorsion), to minimise the distorsion introduced when coding the source (resp. rate).

3.2. Lattice Vector Quantization

The most promising method introduced for the VQ is certainly lattice vector quantization (LVQ) [7], for which the code-book is not calculated. It is defined as a particular subset of regularly arranged points in a k-dimensional space, centered in zero (lattice [8]). When designing a LVQ, the difficulty is not the same as a LBG-type algorithm [9] which has computationally expensive encoding and code-book storage, but lies with the choice of lattice, its truncation and labeling of the remaining points.
In [10, 11], motivated by video coding applications, we introduced the tree-structured lattice VQ (TSLVQ) for which lattice use is easier.

3.3. Tree-Structured Lattice Vector Quantization

The TSLVQ is based on a lattices embedding strategy. Namely a source vector is projected into a first truncated lattice; to get a finer quantization stage, another lower scale truncated lattice is embedded into the Voronoï cell where lies the input vector; this lattice embedding operation can be repeated. Figure 2 illustrates the resulting multi-stages quantization procedure using successive scaling, translating and rounding operators.
We only detail now the TSVQ scheme used in order to compare the 1-D mosaics (for more general information see [11]). The support lattice is $\mathbb{Z}^n$ for which Conway and Sloane determined fast quantizing and decoding algorithms [12] and because it permits an optimal lattice embedding. Considering figure 2 we have:

- $F = (b * \rho)/\sqrt{L_{2\ max}}$ the first scaling factor used to project the input vector $\mathbf{x}$ into the first truncated lattice, $L_{2\ max}$ is the maximal L_2 norm of $\mathbf{x}$, $b = 3$, and $\rho = 0.5$ the packing radius of $\mathbb{Z}^n$;

- $b = 3$: the scaling factor used to project the vector into the next truncated lattice and to get a finer quantization;

- $\mathbf{y}_j$: the reproduction vector of the truncated lattice at the j-th stage.

The final value of the reproduction vector associated with $\mathbf{x}$ will be: $\mathbf{y} = (1/F) * \sum_j (\mathbf{y}_j / b^{j-1})$ where j points out the number of the stages. The code-book has a B-ary tree structure where B corresponds to the number of points of the basic truncated lattice. Each lattice point (i.e each reproduction vector) and its Voronoï cell are associated with a tree node. The 0 vector and the whole source space are associated with the root tree. The possible children of a node are the points of the lattice which is embedded into the parent Voronoï cell. A stage in the tree corresponds with a scale in the lattice hierarchy: the deeper is the tree, the finer is the resolution. The final code-book is the set of terminal nodes or leaves.

Figure 8 (a) illustrates this code-book design. The source

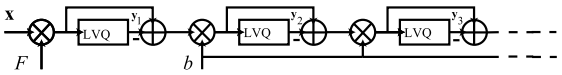


Fig. 2. TSLVQ principle.

vectors are the pixels of a 1-D mosaic. The vector dimension $k = 3$ corresponds to the RGB components of a color pixel. The maximal norm of a vector is then $L_{2\,max} = 3 * 255^2$. The number of points in the basic truncated lattice equals to $B = 3^3$, and the code-book has 3 stages.

4. SCENES COMPARISON

A 1D-mosaic is calculated for each shot of the video. We assume that similar shots have similar 1D-mosaics and our goal becomes to automatically group them. To achieve an efficient and fast comparison between the 1D-mosaics, we handle with their vector quantized representations.

4.1. Code-books comparison

We compare directly the code-books to detect the close 1D-mosaics. Because of the tree-structure of these code-books and the LVQ use, a progressive and fast search can be performed. The comparison method between a "reference" 1D-mosaic with a set of "candidate" 1D-mosaics follows:

- the reference code-book is completely constructed from the reference 1D-mosaic with 3 stages;
- then considering one by one each candidate code-book: its first stage is calculated and compared with the first stage of the reference code-book. If they are close enough, the second stage of the candidate code-book is added and compared with the second stage of the reference code-book. The principle is the same with the third code-books stages.

Because of the time efficiency of this process, a large set of candidates can be handled. Two code-books at the i-th quantization stage are considered close if one of these criteria is satisfied:

- they share a lot of Voronoï cells:

$$NV(i) = \frac{\sharp V_c(i)}{\sharp V_t(i)} \geq NV_{th}(i) \tag{1}$$

  where $\sharp V_c(i)$ is the number of common reproduction vectors between the reference code-book and the candidate, and $\sharp V_t(i)$ is the total reproduction vectors considering the both code-books. For our experiments we use typical thresholds as $NV_{th}(1) = 60\%$, $NV_{th}(2) = 50\%$, and $NV_{th}(3) = 40\%$ for the three successive quantization stages.

- the shared cells contain almost all the vector source:

$$\begin{aligned} NS_{ref}(i) &= \frac{\sharp S_{ref}(i)}{\sharp S_{ref}} \geq NS_{ref_{th}}(i) \quad \text{and} \\ NS_{cand}(i) &= \frac{\sharp S_{cand}(i)}{\sharp S_{cand}} \geq NS_{cand_{th}}(i) \end{aligned} \tag{2}$$

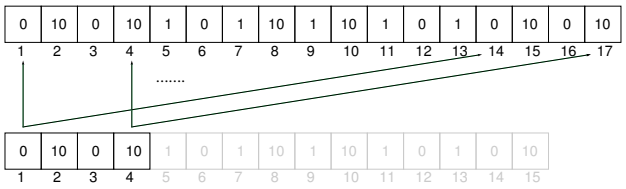
  where $\sharp S_{ref}(i)$ (resp. $\sharp S_{cand}(i)$) is the number of reference (resp. candidate) source vectors inside the common Voronoï cells, and $\sharp S_{ref}$ (resp. $\sharp S_{cand}$) is the total number of reference (resp. candidate) source vectors. We use typically $NS_{ref_{th}}(i) = NS_{cand_{th}}(i) = 50\%$ for $i = 1$, 40% for $i = 2$ and 30% for $i = 3$.

4.2. Reduced mosaics comparison

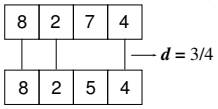
If two code-books are close enough (at a given quantization stage), the next step consists in comparing the colors order between the two corresponding vector-quantized 1D-mosaics. In practice a similarity coefficient is calculated from simplified versions of the 1D-mosaics. The similarity is assessed by using the editing distance principle [13]. The simplified version of a vector-quantized 1D-mosaic is obtained, when scanning from left to right the mosaic, by merging the successive pixels with identical values, i.e. if the full mosaic presents successive identical values, these latters are copied (just once) in the reduced form of the mosaic. Equation 3 details the editing distance implementation between two (vector-quantized and reduced) mosaics blocks:

$$d(b_r, b_t) = \frac{\sharp Identical\, values}{\sharp b_r\, values} \tag{3}$$

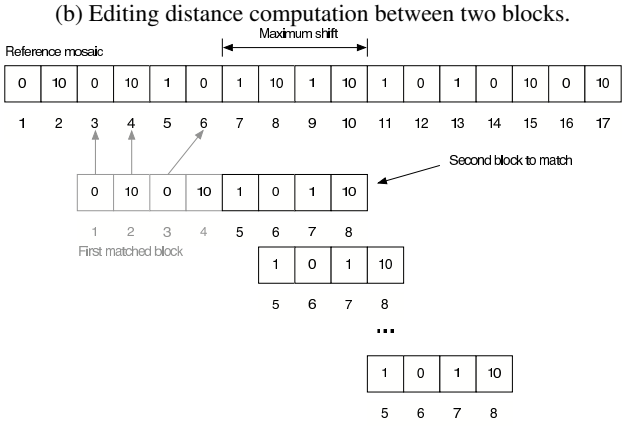
where the editing distance $d(b_r, b_t)$, between the block of pixels b_r of the reference reduced mosaic and a corresponding block of pixels b_t of the reduced tested mosaic, is given by the ratio between the number of identical pixels values and the reference block size. The complete editing distance computation steps (considering all the mosaics blocks) are given in figure 3 with a block length equals to 4 pixels. This



(a) Matching between the tested mosaic (bottom) and the reference block (top).



(b) Editing distance computation between two blocks.



(c) Block matching process.

Fig. 3. Block comparison and editing distance computation.

editing distance computation is performed block by block. For a given block size, once an editing distance is computed between the two first blocks, the process is iteratively repeated to the next blocks on the mosaic's right hand side. In the presented example (see figure 3(c)), when a first reference mosaic block has been successfully matched with a tested mosaic's block (by providing a high editing distance), the next right hand side block is considered. The tested mosaic block is then shifted (pixel by pixel) towards the end of the reference mosaic and an editing distance is computed for each single step. When several successive blocks get a high similarity coefficient, their overall editing distance is computed (mean value of the editing distances), the set of blocks presenting the highest similarity is kept intact while the other sets of blocks presenting a lower mean editing distance are discarded. An important feature of the editing distance in this context lies in its ability to consider both the apparition and suppression of new values in the tested sequences, i.e. for each block comparison, several modified reference blocks are computed and compared, each modified version taking into account a possible value shift. For each block matching process the current block is considered as matched if the corresponding editing distance is above a predefined threshold (empirically set here to 0.6).

5. EXPERIMENTAL RESULTS

We present in this section two distinct results sets (denoted as set A and set B). The first one will present a straightforward case, where only one mosaic clearly appears as the best candidate for a good matching with the reference (Figures 4). In the second set, several mosaics appear to be good candidates to match the reference sequence.

5.1. Code-books comparison

A similarity indicator I has been introduced to represent the classification results:

$$I = \sum_{i=1}^{3} \left(I(i) \times \frac{33}{2} + 33 \times (i-1) \right) \tag{4}$$

where i represents the quantization stage (see also definitions at the equations 1 and 2):

$$I(i) = NV(i) * (NS_{ref}(i) + NS_{cand}(i)) \tag{5}$$

The indicator I aims at classifying the mosaics code-books into 3 regions according to their similarity with the reference mosaic code-book, and considering the 3 (possible) stages of the code-books comparison: the deeper is the code-books comparison, the higher is I.
Figures 4 and 5 illustrate the result for the two considered sets, when comparing 70 candidates 1D-mosaics code-books with a reference mosaic code-book (the shots are from the video sequence "Tympanon", SFRS). Figure 4 clearly shows that only one mosaic code-book matches the reference. The reference mosaic code-book evidently gets a hundred percent rate. On the contrary, in figure 5, many mosaics code-books can be considered as good candidates.

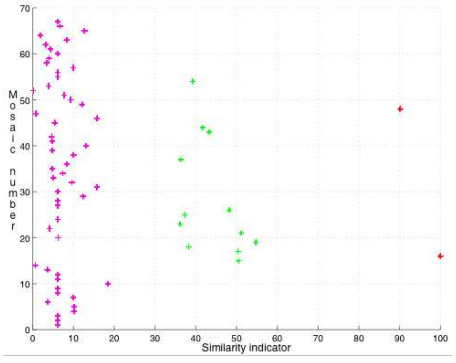


Fig. 4. Comparison of a 1D-mosaic code-book with 70 others (set A).

The two closest 1D-mosaics in figure 4 are shown in figure 6, and their corresponding code-books are represented in figure 8. Considering the second results set (i.e. set B),

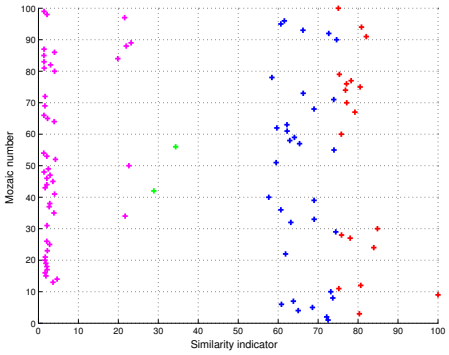
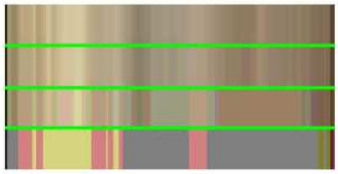
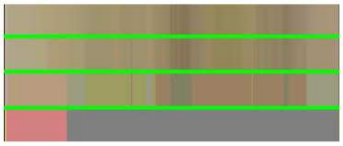


Fig. 5. Comparison of a 1D-mosaic code-book with 70 others (set B).

the reference mosaic is given in figure 7 (a) along with its two closest mosaics, figures 7 (b) and 7 (c), and finally, the corresponding code-books are given in figure 9. Vertical axis in figures 6 and 7 (right sub-figures) represents the different quantized versions of the 1D-mosaics corresponding to the three different stages of quantization (as denoted in fig. 7 (a)). In figures 8 and 9, the three axis represent the Red, Green and Blue pixels' values.



(a) Reference mosaic (set A)



(b) Candidate mosaic (set A)

Fig. 6. TSLVQ of 1D-mosaics: from top to bottom each picture shows the original 1D-mosaic and its quantized version using a code-book with three, two or one stage.

5.2. Reduced mosaics comparison

The code-books comparison achieves a quick sort between the 1D-mosaics : for a reference 1D-mosaic we get a set of candidates. The next step consists in performing a finer comparison between these candidates.

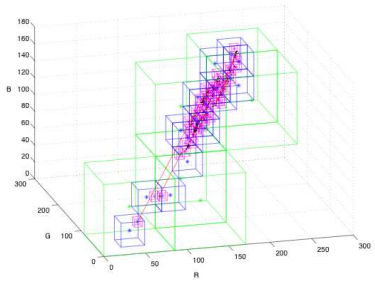
As previously detailed, the vector quantized 1D-mosaics are first reduced in order to be compared. This process is depicted in figure 10. This figure both shows a full 1D-mosaics comparison (10 (a)) and a reduced 1D-mosaics comparison (10 (b)). Each sub-figure shows both the reference 1D-mosaic (top mosaic) and the considered candidate (bottom mosaic). Figure 10 clearly shows how successive pixels of the full 1D-mosaics are merged to provide the reduced

(a) Reference 1D-mosaic (set B)

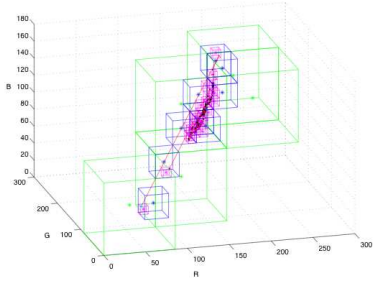
(b) First candidate 1D-mosaic (set B)

(c) Second candidate 1D-mosaic (set B)

Fig. 7. TSLVQ of 1D-mosaics: each sub-figure presents both the motion based compensated 1-D projection (left sub-figure, see figure 1 for details) and its vector quantized versions (right sub-figure).

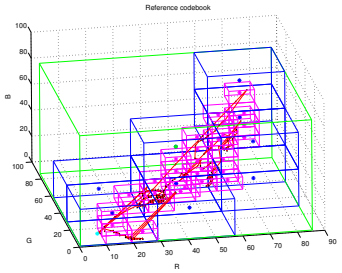


(a) Reference Code-book (set A)

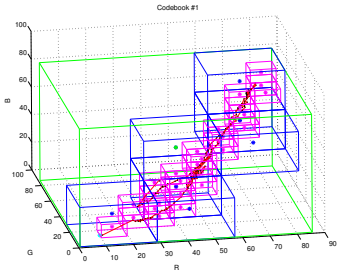


(b) Candidate Code-book (set A)

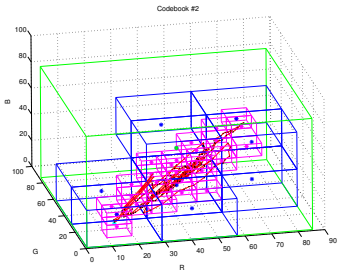
Fig. 8. Code-books design (set A): the source vectors are the black dots, the red line symbolises the 1D-mosaic pixels scan. The Voronoï cells and the reproduction vectors are represented by three different size cubes corresponding to the different quantization stages.



(a) Reference Code-book (set B)



(b) First candidate code-book (set B)



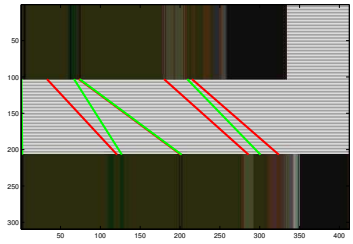
(c) Second candidate code-book (set B)

Fig. 9. Code-books design (set B): the source vectors are the black dots, the red line symbolises the 1D-mosaic pixels scan. The Voronoï cells and the reproduction vectors are represented by three different size cubes corresponding to the different quantization stages.

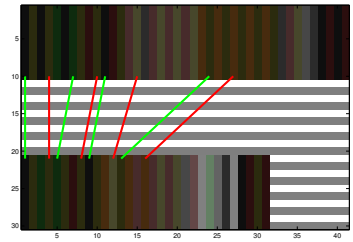
form of the 1D-mosaics. The editing distance computation between these reduced 1D-mosaics leads to determine similar shots. Based on the previously given editing distance computation (with a selection threshold set to 0.6), for a given reference shot (two images of this shot are given in figure 11 (a)), a set of 8 similar shots have been found, see figure 11 (b). Although a few shots appears to be different from the reference, most of the obtained shots closely match the request.

6. CONCLUSION

We have introduced a new tool for grouping video shots into scenes and hyper-scenes. The original method is based on the TSLVQ of 1D-mosaics, the comparison between the code-books followed by a sequences comparison. The tree



(a) Full 1D-mosaics comparison.



(b) Reduced 1D-mosaics comparison.

Fig. 10. 1D-mosaics comparison: a green (respectively red) line symbolizes the beginning (respectively end) of a matched block.

(a) Two images of the reference shot.

(b) Closest shots.

Fig. 11. Video browsing results.

structure of the code-books allows a hierarchical and pro-

gressive search of the similarity. Because of the lattices use, the method is computationally inexpensive. For similar code-books, reduced mosaics are computed and compared by using the editing distance principle. This technique has been successfully assessed here for two video sequences. An evident strength of the presented technique lies in the use of very small mosaics data to efficiently retrieve similar shots among an important video shots set. Due to both its rapidity and the small amount of stored information, the proposed technique is perfectly suitable for important video databases context. Evidently, a more efficient descriptor could be obtained by computing more 1D-mosaics from the video sequence (i.e. different Mojette Projection angles), nevertheless, an important power of the presented results lies in the use of only a very reduced data set (1D-mosaic) to summarize a full video.

ACKNOWLEDGMENTS

The authors wishes to express their thanks to Xavier Batista Casanelles for his works on the graphical user interface of the presented application.

7. REFERENCES

[1] ISO/IEC JTC 1/SC 29/WG 11/M6156. Mpeg-7 multimedia description schemes wd (version 3.1), July 2000.

[2] M. Irani, P. Anandan, and al. Efficient representation of video sequences and their application. Signal Processing: Image Communications, 8:327–351, 1996.

[3] J.P. Guédon and N. Normand. The mojette transform: applications for image analysis and coding. Proc. of Visual Communications and Image Processing VCIP, 3024:1220–1230, February 1997.

[4] W. Dupuy, J. Benois-Pineau, and D. Barba. 1-d mosaics as a tool for structuring and navigation in digital video content. Proc. of the Internationnal Workshop VLBV, pages 93–100, September 2003.

[5] J. Benois-Pineau, W. Dupuy, and D. Barba. Recovering of visual scenarios in movies by motion analysis and grouping spatio-temporal colour signature of video shots. Invited paper at EUSFLAT 2001, special session on Data Mining Multimedia Systems, pages 385–389, septembre 5-7 2003.

[6] A. Gersho and R.M. Gray. Vector Quantization and Signal Compression. Kluwer Academic Publishers, Boston, 1992.

[7] M. Barlaud, P. Solé, T. Gaidon, M. Antonini, and P. Mathieu. Pyramidal lattice vector quantization for multiscale image coding. IEEE Transactions on Image Processing, 3(4):367–381, July 1994.

[8] J.H. Conway and Sloane N.J.A. Sphere Packings, Lattices and Groups, 2nd edition. A series of Comprehensive Studies in Mathematics. Springer-Verlag, New York, 1993.

[9] Y. Linde, A. Buzo, and R.M. Gray. An algorithm for vector quantizer design. IEEE Transactions on Communications, 28:84–95, 1980.

[10] V. Ricordel and C Labit. Vector quantization by packing of embedded truncated lattices. In Proc. of International Conference on Image Processing ICIP, volume 3, pages 292–295. Washington DC, USA, October 1995.

[11] V. Ricordel and C Labit. Tree-structured lattice vector quantization. In Proc. of European Signal Processing Conference EUSIPCO, volume 2, pages 731–734. Trieste, Italie, September 1996.

[12] J.H. Conway and Sloane N.J.A. Fast quantizing and decoding algorithms for lattice quantizers and codes. IEEE Transactions on Information Theory, 28(2):227–232, March 1982.

[13] R. A. Wagner et M. J. Fisher. The string to string correction problem. Communication of the ACM, 20(10), May 1974.